Debreceni Egyetem
Informatikai Kar

Diplomamunka

Szoftverfejlesztés java nyelven
Grafikus megjelenítés automatizálása a Spark nyelvhez

Készítette:
Szeghalmi Endre
Programtervező matematikus
Nappali tagozat

Témavezető:
Espák Miklós

2007 Debrecen

Köszönetnyilvánítás

Szeretném megköszönni a lehetőséget az INES (Institut National de l'Energie Solaire) munkatársainak, akik részt vettek a projekt megvalósításában. Különösképp **Pierre Tittelein**-nek, aki nagy segítségemre volt a Spark programozási nyelv és platform megértésében, **Étienne Wurtz**-nek az INES vezetőjének, a projekt ötletének támogatásáért.

Köszönöm Espák Miklósnak, aki a Debreceni Egyetem részéről volt a témavezetőm és Java programozási ötletekkel is ellátott.

Köszönöm **Laurence Vignollet** tanárnőnek, aki rengeteget segített a Franciaországban eltöltött egyetemi év alatt. Ő volt az, aki átlendített a kezdeti nehézségeken és a diplomamunkám keretén belül szeretném megköszönni azt a tapasztalatot, amelyet az általa vezetett egyetemi projekt során szereztem.

Nagy tisztelettel adózom dr. **Bognár Katalin** és dr. **Juhász István** egyetemi munkásságának. Köszönöm az oktatást a Debreceni Egyetem Informatikai Kar és a Savoie Egyetem Master 2 ISC (Informatique et Systèmes Coopératifs) oktatóinak.

Külön köszönetem fejezem ki **Louis Stephan**-na**k**, **Simon Le Poultier-nek** és **Simon Istvánnak**, amiért részt vettek a speciális makro linkek kapcsolati problémáinak a megvitatásában és a szoftver tesztelésében.

Tartalomjegyzék

1. Bevezetés

Franciaországban a Savoie Egyetemen eltöltött év második felében dolgoztam ki a diplomamunkám a Nemzeti Napenergiai Intézet (l'Institut National de l'Energie Solaire) Optimalizációs Laboratóriumában (LOCIE). 2006 májusa és szeptembere között.

Diplomamunkámban automatizált grafikus megjelenítés megvalósítását mutatom be a Spark programozási nyelvhez. A Spark egy szimulációs környezet, amely lehetővé teszi egyenletrendszerek megoldását algebrai és differenciális módszerekkel. Nagyon hatékony eszköz különféle energetikai problémák modellezésére.

Elsősorban azért választottam ezt a témát, mert kihívást láttam benne, másodsorban pedig nagy szabadságot kaphattam a munkám során. A program ötletét és koncepcióját is én dolgoztam ki. Szakmailag hasznos volt, hogy végig kell vinni egy projektet a kezdeti fázistól egészen a tesztelésig, a fejlesztői dokumentációt elkészítettem angol és francia nyelven.

A munkám során a következő feladatokat végeztem:

- A Spark nyelv és platform lehetőségeinek feltérképezése
- Automatizált grafikus megjelenítés koncepciójának a kidolgozása
- A megfelelő eszközök kiválasztása a projekt megvalósításához
- A szoftver implementálása
- Dokumentáció készítése francia és angol nyelven
- A követendő irányvonalak meghatározása
- A projekthez kapcsolódó specifikus tudás átadása a projektet folytató csapatnak (knowledge transfer)

A továbbiakban bemutatom a projekt kereteit, majd pedig részletesen magát a projektet. Ismertetek néhány, a munkám során felmerülő problémát és azok megoldásait. Végül ismertetem az elért eredményeket és néhány lehetőséget a projekt folytatására.

2. A projekt keretei

2.1. Az INES rövid bemutatása

A diplomamunkámat egy Lyonhoz közeli cégnél készítettem, az INES (Institut National de l'Energie Solaire) Napenergia Kutató Intézetnél. Három részlegből áll: az INES RDI, amely szerepe a kutatás, fejlesztés és innováció (Recherche, Développement et Innovation); az INES Oktatási Részleg (INES Education), melynek feladata a napenergia professzionális hasznosítási módszereinek tanítása az oktatási intézmények részére; az INES Demonstrációs Részleg (INES Démonstration), feladata a különböző projektek megvalósítása, kísérleti épületek építése a maximális energia megtakarítás céljából. A diplomamunkámat az INES RDI keretén belül végeztem.

Az INES három fő kutatási területe a photovoltaikus napenergia, termikus napenergia és ezek épületekbe való integrációja. A munkám a harmadik kutatási csoportba kapcsolódott bele.

2.2. A LOCIE bemutatása

A LOCIE (Laboratoire Optimisation de la Conception et Ingénierie de l'Environnement) az INES RDI alkalmazott technológiákkal és fejlesztésekkel foglalkozik az épületek fundamentumain.

A LOCIE céljai többek között:

- Kísérletek végzése építési anyagokkal
- Fizikai szerkezetek vizsgálata, melynek célja az anyagokból történő legnagyobb energia kinyerése., ezek energiaszintjeinek megőrzése

A LOCIE három operációs csoportra oszlik:

- A környezet minőségének javítása, különféle technikák kialakítása, tartós anyagok fejlesztése

- Az anyagok összetételének és struktúrájának javítása. CMOS (Comportement des matériaux et optimisation des structures)

- Tartós energia fejlesztése és megőrzése DENED (Développement énergétique durable). Ez a részleg felelős a kifejlesztett új technológiák alkalmazásba állításáért. Egyik ilyen jelentős alkalmazás a kísérleti épületek modellezésének a gyakorlati kivitelezése. Ennek során különböző modellek alapján építik meg a tesztépületeket. Ezek az innovatív eljárások többéves fejlesztés eredményein alapulnak.

Ennek a széles spektrumnak köszönhetően tudja megérteni a LOCIE a környezeti problémákat globális szélességben és ennek köszönhetően jelentős pozíciót foglal el Franciaország kutatóközpontjai között.

Munkámat a DENED keretén belül végeztem, az INES Technolac területén található kutatóközpontban.

3. A diplomamunka bemutatása

3.1. A diplomamunka leírása

- **A diplomamunka témája**:

Automatizált grafikus megjelenítés koncepciójának kidolgozása a Spark nyelvhez és annak megvalósítása. A Spark legfontosabb célja a termikus és photovoltaikus jelenségek modellezésének a kivitelezése.[1, 2]

- **A megvalósítandó projekt részletei**:

Az elgondolás az, hogy készítünk egy **grafikus reprezentációt** a Spark nyelvben megtalálható modellek alapján. Ezek a modellek szövegfájlokban találhatók, specifikus kiterjesztésűek, a Spark nyelv programjait képezik. A megvalósítandó feladat két részből áll: a koncepció megalkotása és a megvalósítandó «parser». A parser segítségével olvassuk ki a megfelelő információkat, megállapítva a kapcsolatokat a modellek között. Grafikus módon kell megjeleníteni a modellben résztvevő elemeket és a közöttük fennálló speciális kapcsolatokat. Ezek az automatizáltan előállított képi információk segítségéül szolgálnak az INES-nél folyó kutatásban.

Követelmény a Spark programozási nyelv megértése és egy **grafikus jelölésrendszer** elkészítése. Másodsorban, ezt a grafikus jelölésrendszert felhasználva készíteni kell egy alkalmazást, ami megoldást nyújt a grafikus megjelenítési problémákra. Ezt következően dokumentálni a megvalósított programot és megfelelő fejlesztői dokumentációt átadni a jövőbeli fejlesztések elősegítése végett.

- **Specifikus információ:**

Minden egyes modell szövegfájlokban van tárolva, ami tartalmazza a feldolgozáshoz szükséges információkat. Ezek a megadott információk egy jól definiált nyelv alapján vannak megalkotva.

A kiválasztott programozási nyelv Java és ajánlott felhasználni a következő ún. pluginok-at: EMF (a Spark sajátos nyelvének leírására) és Merlin (képek generálására). Pluginok alatt a fejlesztői rendszerekbe beépíthető további lehetőségeket értjük.

3.2. Miért ez a projekt?

Ez a projekt lehetővé tette számomra, hogy részt vegyek egy nemzetközi kutatási programban. A kutatási program az energia fenntarthatóságával foglalkozik az épületekben, termikus és aerodinamikai szempontból. A projekt neve DYNASIMUL és 1,9 millió eurós támogatással rendelkezik, amelyet különböző Európai Uniós forrásokból pályázott, valamint a Nemzeti Kutatásért Ügynökség (l'Agence National pour la Recherche) bocsátott a projekt finanszírozására.

A projekt célja, hogy alapjaiból létrehozva kialakítson egy platformot a kísérleti épületek megalkotásához, ahol szimulációkat hajthatunk végre. A bemenő adatok alapján megkapható eredmények számolásához hasznos a Spark. A Spark programozási nyelv, alkalmazás és nagyon hatékony platform is egyben, mely algebrai és differenciális módszerekkel dolgozik. Hátránya, hogy hiányzik az **automatizált grafikus megjelenítés**. Emiatt jelentős idő szükséges egy már működő Spark platform alatt megírt kódrészlet, projekt megértéséhez és bonyolult a kapcsolatok nyomon követése egy már megírt alkalmazás elemei között. Hatékony grafikus megjelenítés segítségével csökkenthető a tanulási idő.

Végeredményben grafikusan prezentálni a már meglévő szimulációkat. Ezek után pedig specifikációt adni egy grafikus interfész megvalósításához.

3.3. A Spark mint platform és alkalmazás

A Spark a nyelvi definíciók mellett egy szimulációs környezet és platform is egyben, amely lehetővé teszi egyenletrendszerek megoldását algebrai és differenciális módszerekkel. Fejlesztése a Simulation Research Group (Szimulációs Kutatócsoport) által történt. Ez a kutatócsoport a

Lawrence Berkeley National Laboratory (California, USA) egyetemi kutatóközponjához tartozik és közvetlen kapcsolatban áll az IBM Los Angelesi kutatócsoportjával.

Egy probléma megoldásához a Sparkban le kell bontanunk a rendszerünket objektumorientált módszerekkel. Komponenseket képezünk az egyenletrendszereinkből. A komponensek speciális egyenletrendszereket tartalmaznak és ezek alkotják a modelleket. Minden egyes speciális egyenletrendszer (vagy egyenlet) objektumként van leképezve a Sparkban. Ahhoz, hogy egy újabb modellt, alkossunk felhasználhatjuk eddigi modelljeinket, tehát fontos az újrahasznosíthatóság fogalma. A probléma jobb megvilágítására szemléltetni fogok egy egyszerű példát, amelynek segítségével jobban megismerhetjük a Spark sajátos nyelvezetét.

A modellek sajátosságai, hogy parametrizáltak, ezáltal nagyfokú skálázásra adnak lehetőséget. A komplex egyenletek és egyenletrendszerek termodinamikai és aorodinamikai modelleket jellemeznek.

Példa egyenletrendszerre:

Egy négyzetméternyi falon áthatoló hő mennyiségének meghatározása a két fal oldalán lévő hőmérsékletkülönbség és fal fizikai szerkezete ismeretében.

Elképzelhetjük egy teljes épületet mennyi egyenletrendszerrel szükséges jellemeznünk. Az egyenletrendszerek szoros kapcsolatban állnak. A kapcsolatok feloldására a Spark speciális linkeket vezet be. A linkek kapcsolják egymáshoz a modelleket és a kapcsolatokat viszonylag bonyolult megérteni a Sparkban.

Az objektum a Spark nyelvben egy osztályból példányosított egyenlet (bizonyos esetekben egyenletrendszer) példány. A jobb megértés érdekében a diplomamunka keretein belül osztályokról beszélek. Az egyenletekből alkotott osztályok neve: **atomi osztályok**, a kiterjesztésükről .cc ismerjük fel őket. Egy osztály egy másik osztályhoz a változóin keresztül kapcsolódik, ennek a speciális neve **port**. Az osztályok közötti kommunikáció tehát elsősorban

portokon keresztül zajlik. A portok közötti speciális kapcsolódási változók neve **link**.

A Spark nyelv programozásához, az atomi osztályokat össze kell kapcsolnunk a portokon keresztül. Ha magasabb szinten szeretnénk összekapcsolni az osztályainkat újabb típusú osztályt kell bevezetnünk, a **makro osztályok**at. Kiterjesztésük: .cm. Ezek az osztályok is rendelkeznek portokkal, ugyanúgy kapcsolódhatnak az atomi osztályokhoz. A magasabb absztrakció eléréséhez tartalmaznak **makro portok**at Segítségükkel kapcsolódnak egymáshoz a makro osztályok **makro linkek**en keresztül. A makro linkek tekinthetők a linkek speciális szempontok szerinti összegzésének. Egy szimuláció jobb áttekinthetőségét, strukturálását teszi lehetővé a makro linkek megléte. Feltétel viszont, hogy egy makro osztály atomi osztályhoz csakis egyszerű linkekkel kapcsolódhat.

A fentebb taglalt fogalmak segítségével építhetjük fel a modelljeinket, az egyszerű egyenletektől eljutva egészen a bonyolult rendszerekig. Az objektumorientált megközelítés nagyon komplex rendszerek felépítését teszi lehetővé. Például egy teljes termikus modellezését egy épületnek.

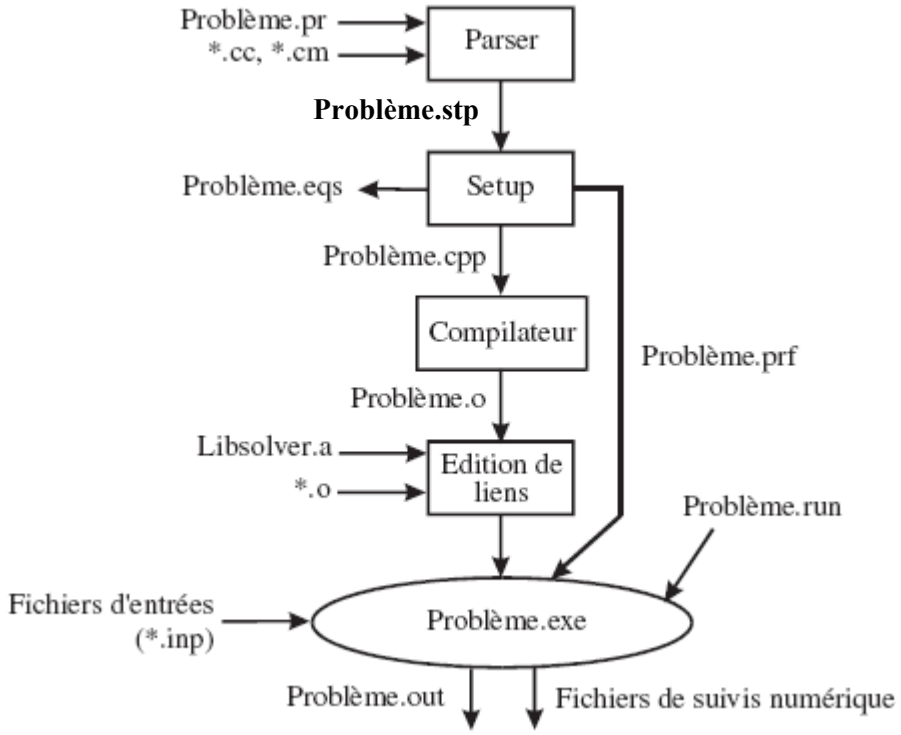
A megtervezett rendszer alapján egy újabb osztálytípust szükséges definiálni, amely segítségével kérdéseket lehet feltenni az általunk megépített rendszerhez. Az osztálytípus neve a **programosztály**. Segítségével lehet megadni a rendszer bemenő paramétereit, és ugyanitt tudjuk definiálni, milyen értékekre vagyunk kíváncsiak. Az ilyen osztály kiterjesztése .pr.

A programosztályok, makro osztályok és atomi osztályok fa struktúrába szerveződnek. A programosztály helyezkedik el a fa struktúra gyökerében.

Egy Spark program lefutásának menete a következőképpen zajlik. Először is beolvasásra kerül egy .pr típusú programosztály. Ezek alapján visszakereződnek a makro osztályok, és a makro osztályok alapján visszakereződnek az atomi osztályok. Az osztályok feldolgozása után, egy működőképes C++ program generálódik. A program bemenő paraméterei a *.inp típusú fájlokban találhatóak. Innen kerülnek kiolvasásra a bemenő

adatok a szimuláció részére. Majd pedig egy problem.out fájl tartalmazza a kinyert információkat. Használunk még egy leíró fájlt: a problem.run osztályt. It a kísérlet lefolyásakor specifikus adatok találhatók, például a szimuláció kezdeti időpontja, a szimuláció végső időpontja és egyéb szükséges paramétereket állíthatóak be.

Az 1. ábra tartalmazza a szükséges lépéseket egy szimuláció lefutásához. A parser a .cc, .cm és a .pr típusú állományokat beolvassa. Kinyeri belőlük a szükséges információkat és egy C++ -ban megírt futtatható állományt generál. Ezt elindítva kapjuk meg az általunk kért eredményeket.



1. ábra: Probléma felépítés a Spark segítségével

3.4. A grafikus megjelenítés automatizálásának a követelményei

A grafikus reprezentáció elkészítéséhez szükséges a Spark programnyelv alapos megismerése. Az első lépés tehát, hogy készítsek Sparkban megírt szimulációkat, majd próbáljam őket ábrázolni a meglévő technikák segítségével. A végső cél a grafikus megjelenítés automatizálása. A

következőkben kódrészletekkel és ábrákkal mutatom be 4.3 fejezetben tárgyalt fontosabb fogalmakat.

A 2. ábrán egy egyszerű programosztály részlete látható.

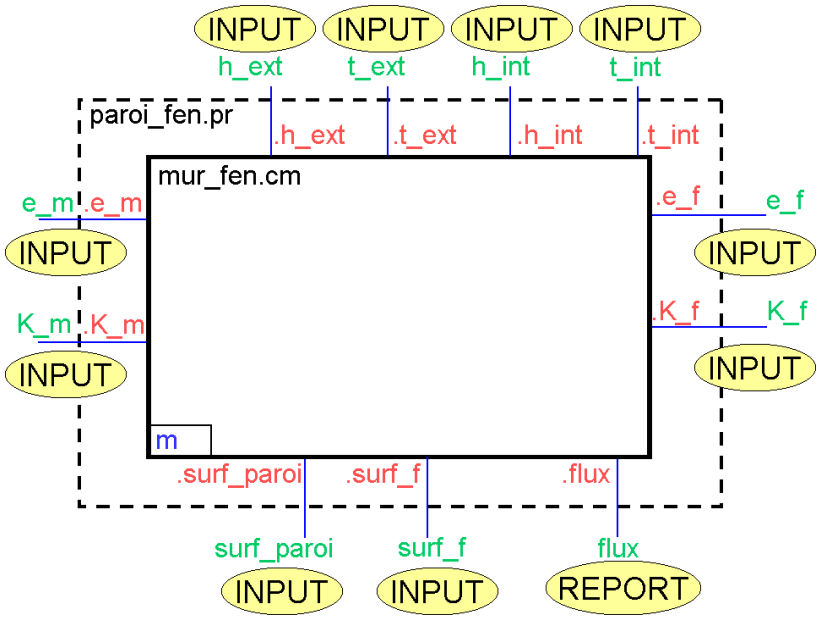
```
1  DECLARE mur_fen m;
2
3  LINK   surf_paroi  m.surf_paroi  "surface totale de la paroi"  INPUT;
4  LINK   surf_f      m.surf_f      "surface de la fenêtre"   INPUT;
5  LINK   flux        m.flux        "flux de chaleur total à travers la paroi"      REPORT;
6  LINK   e_m         m.e_m         "épaisseur du mur"  INPUT;
7  LINK   e_f         m.e_f         "épaisseur de la fenêtre"  INPUT;
8  LINK   K_m         m.K_m         "conductivité du mur"  INPUT;
9  LINK   K_f         m.K_f         "conductivité de la fenêtre"  INPUT;
10 LINK   h_int       m.h_int       "coefficient de convection à l'intérieur"  INPUT;
11 LINK   h_ext       m.h_ext       "coefficient de convection à l'extérieur"  INPUT;
12 LINK   t_int       m.t_int       "température à l'intérieur"  INPUT;
13 LINK   t_ext       m.t_ext       "température à l'extérieur"  INPUT;
14
```

2. ábra: «.pr» kiterjesztésű programfájl

A 3. ábrán látható a kódhoz tartozó képi reprezentáció. A programosztályban deklarálunk egy makro osztályt, majd pedig definiáljuk a közöttük lévő kapcsolatokat. Linkek definiálásának szintaktikája:

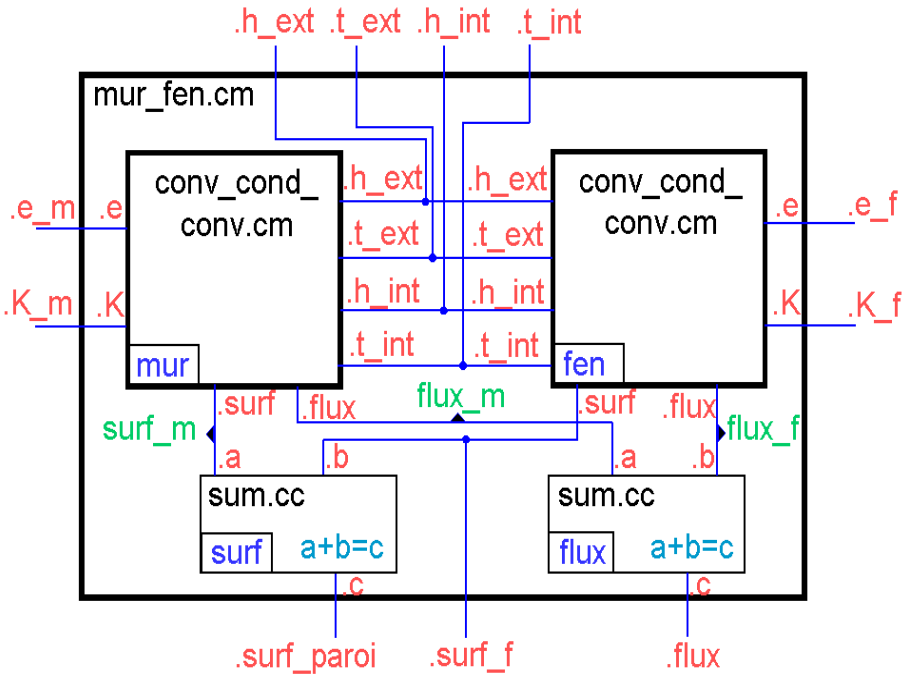
LINK [név] [osztály$_1$.] elem$_1$, [[osztály$_2$.] elem$_2$] … [INPUT,] [OUTPUT]

Ezek a linkek egyszerű atomi linkek. A teljes és részletes makro linkekkel kiegészített szintaktika megtalálható a Spark dokumentációjában [3].



3. ábra: Kézzel készített grafikus reprezentáció «.pr»

A 4. ábrán láthatunk (.cm) kiterjesztésű osztályt. Az osztályok tartalmazási viszonyban vannak egymással. Mindig egy osztályban deklaráljuk a többi osztályt, így tartalmazási viszony alakul ki. Szülő–gyerekosztály viszonnyal, ezek láncolatával, fa struktúrával jellemezhető.



4. ábra: Egy .cm kiterjesztésű osztály grafikus reprezentációja (mur_fen.cm)

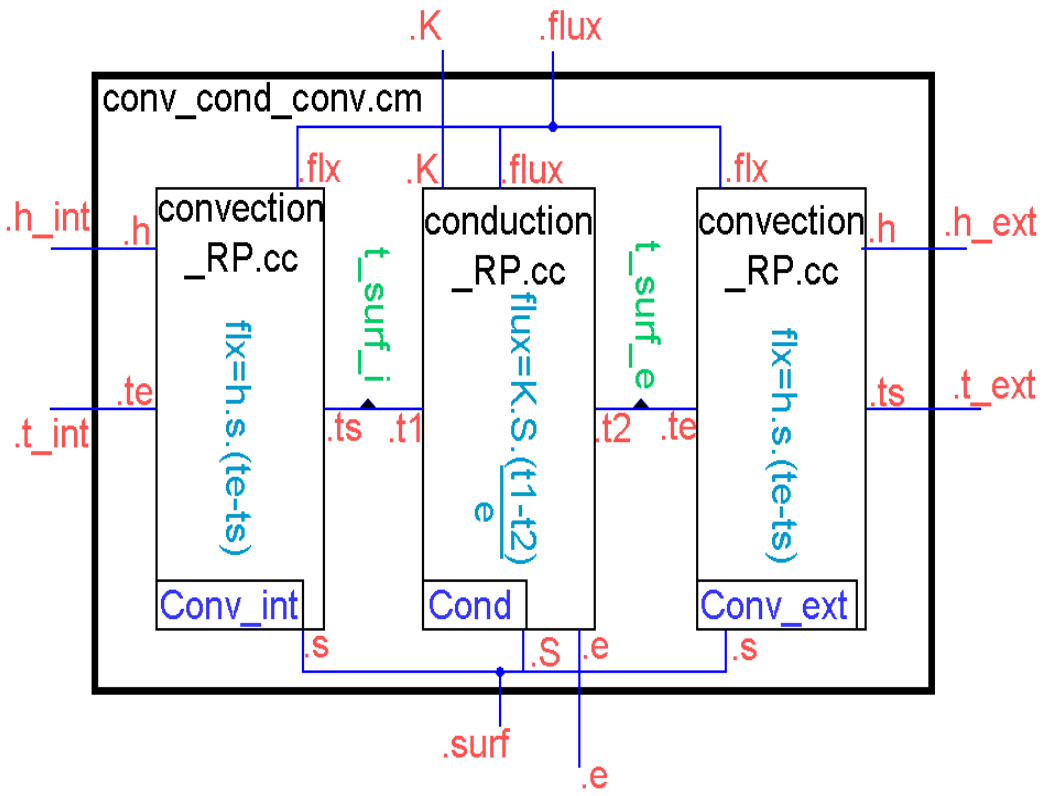
```
/* mur_fen.cm

---Date de création : 17/07/2006
---Dernière modification : 17/07/2006

---Personne ayant créé l'objet : Pierre Tittelein (pierretittelein@yahoo.fr)

---Que fait l'objet : transfert de chaleur à travers une paroi avec fenêtre.
Objet créé pour exemple de représentation graphique.
---Références utilisées :   classique
*/

PORT    surf_paroi  "surface totale de la paroi" [m^2];
PORT    surf_f      "surface de la fenêtre" [m^2];
PORT    flux        "flux de chaleur total à travers la paroi" [W];
PORT    e_m         "épaisseur du mur" [m];
PORT    e_f         "épaisseur de la fenêtre"   [m];
PORT    K_m         "conductivité du mur" [W/(m.deg_C)];
PORT    K_f         "conductivité de la fenêtre" [W/(m.deg_C)];
PORT    h_int       "coefficient de convection à l'intérieur"   [W/(m^2.deg_C)];
PORT    h_ext       "coefficient de convection à l'extérieur"   [W/(m^2.deg_C)];
PORT    t_int       "température à l'intérieur" [deg_C];
PORT    t_ext       "température à l'extérieur" [deg_C];

DECLARE     conv_cond_conv  mur;
DECLARE     conv_cond_conv  fen;
DECLARE     sum             surf;
DECLARE     sum             flux;

LINK    surf_m      mur.surf,surf.a     "surface réelle du mur";
LINK    .surf_paroi surf.c              "surface totale de la paroi";
LINK    .surf_f     fen.surf,surf.b     "surface de la fenêtre";
LINK    .flux       flux.c              "flux de chaleur total à travers la paroi";
LINK    flux_m      mur.flux,flux.a     "flux de chaleur total à travers la paroi";
LINK    flux_f      fen.flux,flux.b     "flux de chaleur total à travers la paroi";
LINK    .e_m        mur.e               "épaisseur du mur";
LINK    .e_f        fen.e               "épaisseur de la fenêtre";
LINK    .K_m        mur.K               "conductivité du mur";
LINK    .K_f        fen.K               "conductivité de la fenêtre";
LINK    .h_int      mur.h_int,fen.h_int "coefficient de convection à l'intérieur";
LINK    .h_ext      mur.h_ext,fen.h_ext "coefficient de convection à l'extérieur";
LINK    .t_int      mur.t_int,fen.t_int "température à l'intérieur";
LINK    .t_ext      mur.t_ext,fen.t_ext "température à l'extérieur";
```

5. ábra: A grafikus reprezentáció forráskódjának egy részlete: «mur _fen.cm»

Az 5. ábra a 4. ábrán található képi reprezentáció lekódolt része. Ezt elemezve láthatjuk, hogy a makro osztály tartalmaz négy újabb osztályt, két makro osztályt és két atomi osztályt. Az osztályok számának a növelése esetén csökken az olvashatóság, romlik a képi információ felhasználhatósága. Elkülönül az ábrázolása a makro osztályoknak (« mur » és « fen »), és az atomi osztályoknak (surf és flux). A példánydeklaráció szintaktikája:

DECLARE OSZTÁLYNÉV PÉLDÁNYNÉV$_1$, [PÉLDÁNYNÉV$_2$]…

A következő két ábrán (6. ábra, 7. ábra) látható egy csak atomi osztályokat tartalmazó makro osztály képi és kódszintű megjelenítése. Mindkét ábra ugyanaz az információ megjelenítését teszi lehetővé, más-más módon.



6. ábra: Egy csak atomi osztályokat tartalmazó makro osztály

```
/* conv_cond_conv.cm

---Date de création : 2/02/2006
---Dernière modification : 17/07/2006

---Personne ayant créé l'objet : Pierre Tittelein (pierretittelein@yahoo.fr)

---Que fait l'objet : transfert de chaleur en régime permanant dans une
paroi avec convection.
Objet créé pour exemple de représentation graphique.
---Références utilisées :   classique
*/
//déclaration des ports

PORT    surf    "surface de la paroi" [m^2];
PORT    e       "épaisseur de la paroi"   [m];
PORT    flux    "flux de chaleur à travers la paroi" [W];
PORT    K       "conductivité de la paroi" [W/(m.deg_C)];
PORT    h_int   "coefficient de convection à l'intérieur"   [W/(m^2.deg_C)];
PORT    h_ext   "coefficient de convection à l'extérieur"   [W/(m^2.deg_C)];
PORT    t_int   "température à l'intérieur" [deg_C];
PORT    t_ext   "température à l'extérieur" [deg_C];

//déclaration des objets

DECLARE conduction_RP cond;
DECLARE convection_RP conv_int,conv_ext;

//liens entre les ports

LINK    .surf       cond.S,conv_int.s,conv_ext.s;
LINK    .e      cond.e;

LINK    .flux       cond.flux,conv_int.flx,conv_ext.flx;

LINK    .K      cond.K;
LINK    .h_int  conv_int.h;
LINK    .h_ext  conv_ext.h;

LINK    .t_int  conv_int.te;
LINK    t_surf_i    conv_int.ts,cond.t1;
LINK    t_surf_e    conv_ext.te,cond.t2;
LINK    .t_ext  conv_ext.ts;
```

7. ábra: A csak atomi osztályokat tartalmazó makro osztály forráskódja

A diplomamunkámban látható miképpen tettem lehetővé a Spark megjelenítésének az automatizálását. A Spark jobb megértése érdekében egy egyszerű példával szemléltetem a Spark lehetőségeit.

3.5. Példa szemléltetése

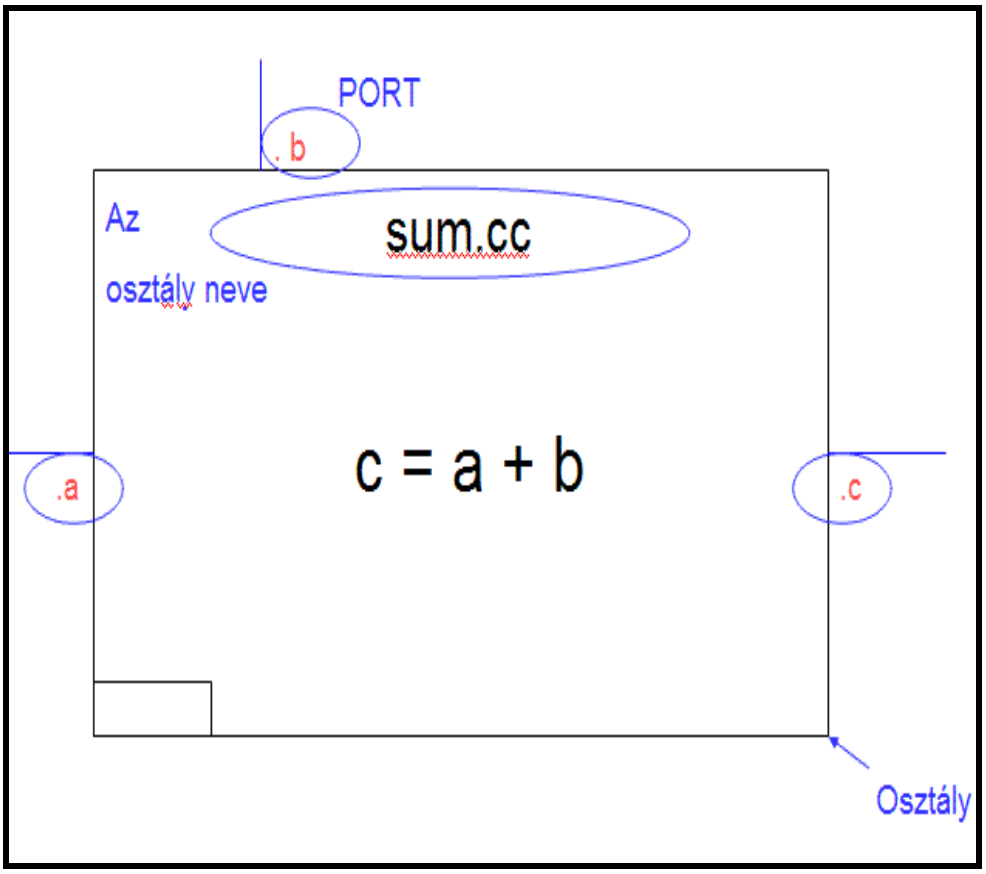
Vegyünk egy egyenletrendszert, amelyet meg szeretnénk oldani.

$$x1 + x2 = x5$$

$$x3 + x4 = x6$$

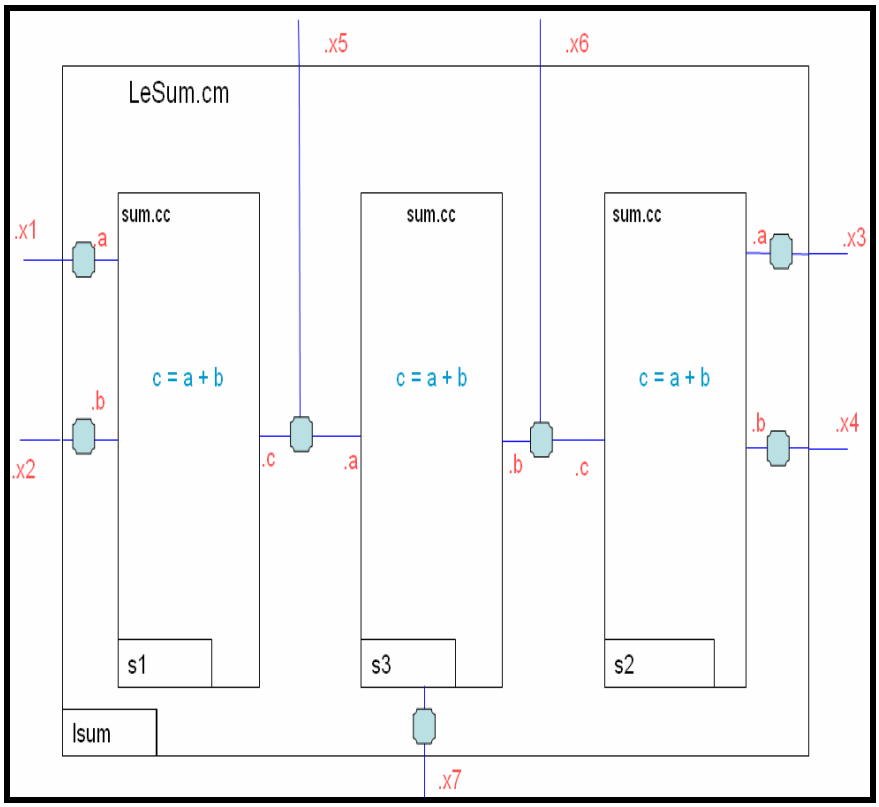
$$x5 + x6 = x7$$

Forráskódok nélkül bemutatom a következő ábrákon az egyenletrendszer megoldását. A Spark honlapján szemléletes dokumentációk és részletes forráskódok találhatók ebben a témában.



8. ábra: Atomi osztály

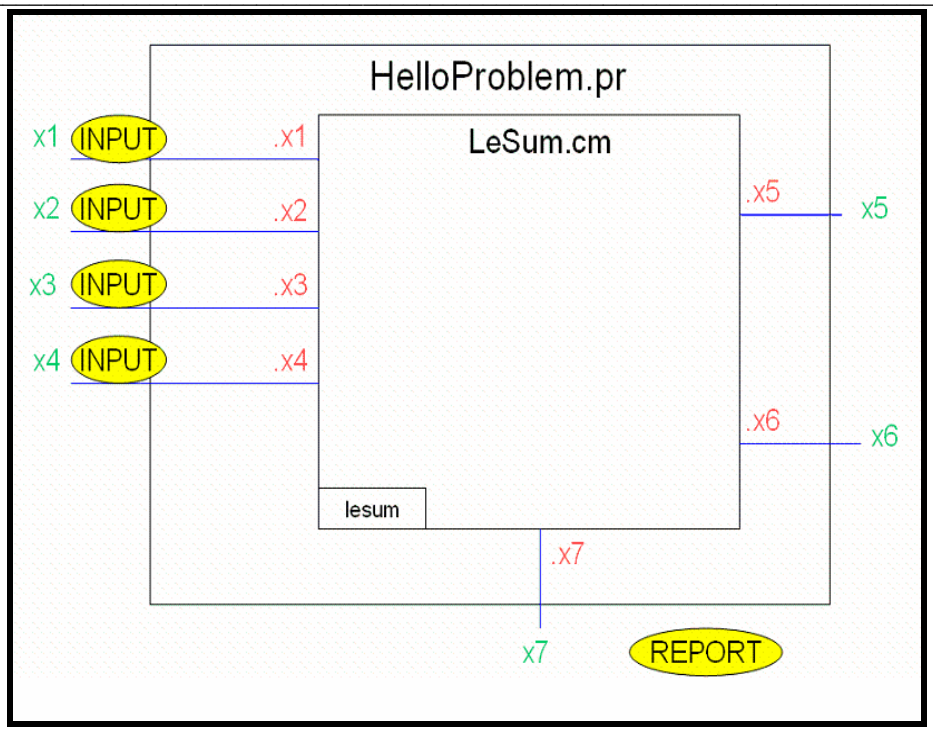
A 8. ábrán az összes **c = a + b** típusú egyenletek halmazát kapjuk meg, megfelelő implementáció esetén felhasználhatjuk **a = b – c** és **b = a – c** típusú egyenletek megoldásaként is. Tehát a 8. ábra egy atomi osztályt ábrázol, ami az összes **c = a + b** egyenletek halmazaként fogható fel.



9. ábra: Makro osztály

Az előbb definiált atomi osztályt felhasználva készítjük el az egyenletrendszert a 9. ábrán. Amint látható, linkekkel kapcsolja össze, a Spark a különböző egyenleteket. Nagyon bonyolult rendszereknél a módszer ugyanaz marad, csak az elemek száma növekszik.

Már csak definiálni kell a programosztályt. Megadva a bemenő adatokat és a kérdést megkapjuk a válaszainkat a szimulációval kapcsolatban. A 10. ábrán a minimális nehézségű példánk egy esetleges programosztálya szerepel.



10. ábra: A programosztály

A programosztály 4 bemeneti paraméterrel rendelkezik, és az x7-ben kapjuk meg az egyenletrendszer megoldását. Ehhez hasonló módszerekkel építhetőek fel tetszőleges méretű komplex rendszerek. Az összes ilyen komplex rendszer a programosztályok, makro osztályok és atomi osztályok összességéből áll.

4. A probléma megoldásának szakaszai

4.1. A Spark megismerése

A munkám elejét a Spark megismerésével töltöttem. Átnéztem a dokumentációkat, és értelmeztem a szoftver sajátosságait. Próbáltam megérteni a komolyabb feladatokat és a szoftver komplexebb lehetőségeit. Ezek után készítettem grafikus reprezentációkat, kézzel vagy valamilyen grafikus segédprogramot segítségül véve.

A szimulációk és a grafikus reprezentációk készítésekor szembe találkoztam ugyanazokkal a problémákkal, mint a Spark programozók.

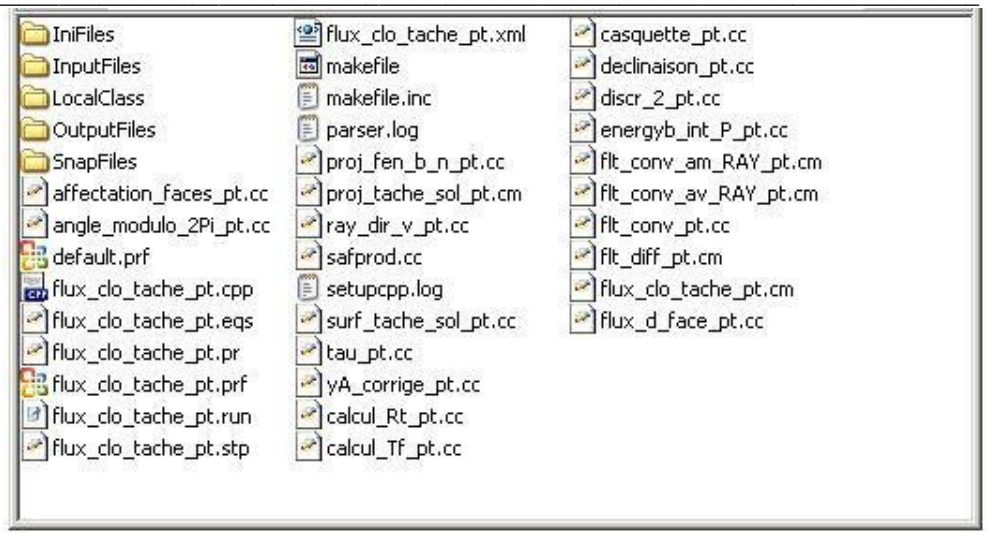
Felhasználói esetek problémaköre:

- Ahhoz, hogy megtalálhassak egy változóhoz (INPUT, REPORT, INIT) tartozó információt nagy mennyiségű osztályt kell, hogy megnyissak.

- Ha rendelkezem egy kész komponenssel, és valamilyen oknál fogva módosítanom kell rajta, nincs megadva az osztályhierarchia. Keresgélni kell egyik osztályról a másikra, hogy a módosítást elvégezhessem manuálisan.

- Nagyszámú osztályt tartalmazó modell esetén nehéz megérteni és ábrázolni az osztályok közötti kapcsolatokat.

- Nagyszámú osztályt tartalmazó modell esetén csökken a láthatóság, és bonyolult az osztályok közötti kapcsolatok átalakítása, az osztályok mozgathatósága.



11. ábra: Spark projekt

- A 11 ábrán látható, egy szimuláció gyökérkönyvtára. Ha a kísérlet több mint 8 osztályt és 15 linket tartalmaz, a képi információ bonyolulttá válik és nehezen értelmezhető. Ezáltal két lehetőség közül tudok választani a feladatom megoldásához.

  - Az első lehetőség, hogy az osztályok (.pr, .cm, .cc) alapján generálok egy alaphelyzetet. Drag&Drop segítségével a felhasználó tudja mozgatni az objektumokat. A probléma ezzel az esettel, hogy a felhasználók olyan projekteket is kell, hogy generáljanak, ahol több mint 30 osztály és több mint 100 link és makro link található. Ilyen esetben a felhasználó nem látja át az optimalizációs lehetőségeket a kép túl bonyolulttá válik és nagyon könnyen belebonyolódhat az objektumainak elrendezésébe.

  - A második lehetőség, találni egy olyan eszközt, amely tartalmaz objektumok elhelyezésére optimalizációs lehetőségeket. Így növelhető a felhasználható osztályok és a közöttük lévő kapcsolatok száma.

4.2. A pluginok feltérképezése

Miután kellően megismertem a Spark nyelvet, továbbléptem a következő fázisra, megvizsgáltam, miképpen használhatnám fel az EMF és a Merlin plugint a grafikus megjelenítéshez.

Ezek a pluginok voltak feltüntetve követelményként, hogy használjam a feladatom megoldásához.

EMF (Eclipse Modelling Framework) egy keretrendszer, modellek alapján Java kód generálható A modellek létrehozhatóak XML segítségével, de akár Java nnotation-ök segítségével is. [7]

A Merlin plugin ingyenes Eclipse plugin amely, az EMF-re támaszkodik, és grafikus generálásra ad lehetőséget.[6]

Áttanulmányozva őket, rájöttem, hogy nem elegendőek a projekt céljaira, mert nem tartalmazzák azokat a lehetőségeket, amelyek hasznosak a szimulációk megjelenítésére. Ekkor kerültem abba a helyzetbe, hogy nem tudtam milyen eszközöket használjak a továbbiakban a projekthez, és milyen módszert fogok használni az információk feldolgozásához.

Tanácsot kértem a SPARK készítőitől, a Berkeley egyetem munkatársaitól. Benne vázoltam a követelményeket, a lehetőségeket és útmutatást kértem a további lépésekhez. Egyetértettek velem, hogy ezek a pluginok nem használhatóak, de további hatékony útmutatóval ők sem tudtak szolgálni.

4.3. Eszközök kiválasztása a projekthez.

Miután rájöttem, hogy nem megfelelő az EMF-Merlin plugin kettős jar csomag, próbáltam más lehetőségek után kutatni a projekt megvalósításához.

Íme azok csomagok, amelyeket használni akartam, és interneten kutattam utánuk Különféle jar csomagokat, illetve nyílt forráskódú és ingyenes szoftvereket vettem figyelembe.

GTK+	A GTK+ segítségével eGUI-kat (Graphical Unit Interface) generálható. Egy teljes csomagot kínál kisebb projektek grafikai interfészének a megvalósítására
Graphviz	A Graphviz egy nyílt forráskódú grafikai vizualizációs eszköz. Több lehetőséget is kínál a grafikai megjelenítésre. Saját nyelvvel rendelkezik, és beépített funkcióval szabványos képek generálhatók.
Jgrapht + Jgraph	JgraphT egy nyílt forráskódú java csomag, megvalósít számos gráfelmélettel kapcsolatos algoritmust. Matematikai hátteret ad a Jgraph-nak. Jgraph szintén nyílt forráskódú java csomag, együttesen használható a Jgraph-al. [4, 5]
JUNG (Java Universal Network/Graph	JUNG egy a Jgraph-hoz hasonlóan nyílt forráskódú grafikus java csomag. Remek vizualizációs, elemző és grafikus képességekkel rendelkezik.
Qucs (Quite Universal Circuit Simulator)	Qucs egy áramköröket tervező szimulációs eszköz. Jól felépített grafikus csomag, parametrizálni lehet a különféle eseteket, és optimalizációs lehetőségekkel is rendelkezik.

A lehetőségek feltérképezése után több lehetőségem lett a projekt megvalósítására. Mindegyiket kipróbáltam, átnéztem a példaprogramjait és teszteltem a funkcionalitásait, hogyan generálhatnék automatikusan képeket a segítségükkel.

Végül lecsökkentettem a számításba vehető lehetőségeket.

- Graphviz
- Jung
- JgraphT + Jgraph

A GraphViz problémája a következő: miután kigeneráltam egy alap pozíciót, nehézkes volt a módosítása. Ráadásul konvertálni kell a Spark programokból kinyert információkat, bemenő paraméterekké a Graphiz nyelvére.

A JUNG jól strukturált java csomag grafikus vizualizáció szempontjából. Viszont nehézkes elkülöníteni a matematikai modellt a grafikus reprezentációtól.

Ezek az érvek alapján választottam ki a JgraphT + Jgraph csomagokat. A JgraphT segítségével fel tudom építeni a matematikai modellt. A matematikai modell alapján pedig meg tudom jeleníteni az információkat a Jgraph-ban. A Jgraph rendelkezik még egy csomaggal, ami fizetős, viszont térítésmentesen használható egyetemi projektekben. A matematikai modell felépítése után a Jgraph segítségével alkalmas eszközt találtam az osztályok és a közöttük lévő kapcsolatok megjelenítésének az optimalizálására. A java csomagnak a segítségével bővíthető a vizualizációban résztvevő osztályok és linkek száma.

4.4. Grafikus megjelenítés automatizálásának a koncepciója

Ahhoz hogy egy jobb grafikus reprezentációt készíthessek, megbeszéléseket folytattam a Spark programozóival. Elemeztem a felhasználói igényeket kérdések feltételével, szimulációkban való résztvétellel és további kézzel készített grafikus reprezentációk készítésével.

A Spark programozókkal való többszörös megbeszélés után elhatároztam, hogy a következő módszerekkel alakítom ki a grafikus megjelenítést. (12. ábra)

Objektumfa kiegészítő információkkal (fizikai elhelyezkedés, tartalmazott linkek stb.)	Egy osztály bemutatása, kezdő pozícióval, és a változtatás lehetőségeivel.
Az adott elem forráskódja («.pr», «.cc», « .cm »)	

12. ábra Az ablakok koncepciója

Az ügyféllel megbeszéltem az ablakok kialakítását tehát folytathattam a következő szakasszal.

4.5. Az objektumok kinyerése

A „parsing” vagyis a források értelmezése megfogalmazásához a következő kérdéseket kell feltenni: Hogyan tudom kinyerni az információkat abból a projektből, amely modellekből, komponensekből, osztályokból áll és a Spark nyelvben van implementálva?

A Spark sok lehetőséget biztosít arra, hogy az osztályokat kapcsolatokkal lássa el. Ha egyesével szeretném kinyerni az információkat a «.cc», «.cm» és «.pr» kiterjesztésű osztályokból, akkor újra kellene, hogy írjak egy Spark értelmezőt (parsert). Miután konzultáltam a Spark programozóival, találtam .stp kiterjesztésű állományokat, amelyek mindig létrehozódnak a szimuláció futásakor. Az .stp kiterjesztésű állományokat egyszerűbb értelmezni, és nem kell keresgélni az operációs rendszerben.

A 13. ábrán egy részlet látható egy «.stp» : kiterjesztésű fájlból.

```
26148     2  // connections
26149     //obj       port        break match
26150     ray_1`ray_0fen_2`rhoS   c       5 5
26151     ray_1`plan_bh_0_0_1`ref a1      5 5
26152 // Link(1) :  surf_i_bh_0_0_2.rhoS  "demo_5_fen_v3_pt.pr", line 149
26153 //  Conn(1)m: ray_1.surf_0fen_2.rhoS
26154 // Link(2) :  ray_1~NONAME115.rhoS  "./LocalClass/rayon_5_fen_pt.cm", line 314
26155 //  Conn(2)m: ray_1`plan_bh_0_0_1.surf_i_1.rhoS
26156 //  Conn(2)m: ray_1`ray_0fen_2.surf_i.rhoS
26157 //  Conn(2)m: ray_1.surf_0fen_2.rhoS
26158 // Link(3) :  ray_1`ray_0fen_2~NONAME11 "./LocalClass/flux_ray.cm", line 59
26159 //  Conn(3) : ray_1`ray_0fen_2`rhoS.c
26160 //  Conn(3)m: ray_1`ray_0fen_2.surf_i.rhoS
26161 // Link(3) :  ray_1`plan_bh_0_0_1~NONAME14  "./LocalClass/surface_1_1.cm", line 40
26162 //  Conn(3) : ray_1`plan_bh_0_0_1`ref.a1
26163 //  Conn(3)m: ray_1`plan_bh_0_0_1.surf_i_1.rhoS
26164 ray_1~surf_bh_0_0_1.ExOS    -       0   0   0.01    -100000 100000  1.E-6  0
26165     6  // connections
26166     //obj       port        break match
26167     ray_1`enc_bh_0_0_0`exis ExOS_5      5 5
```

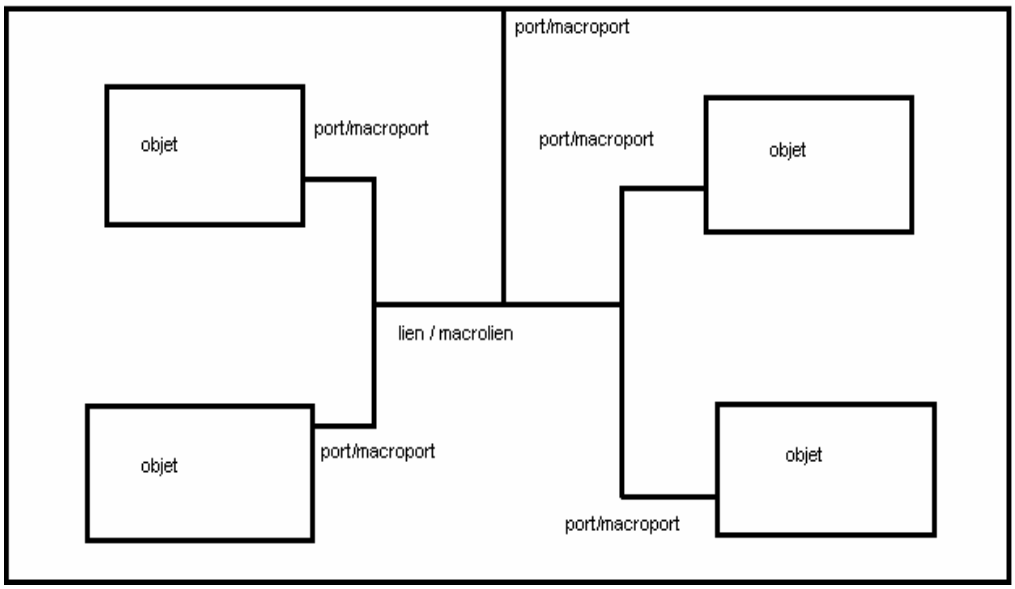
13. ábra Az .stp fájlban található információk értelmezése

.Ahogy az ábrán látható példából is kiderül, egy ilyen (.stp) kiterjesztésű fájl mérete rendkívül nagy lehet. A kapcsolatok gyorsabb értelmezése miatt sokat dolgoztam a memória minél jobb kihasználtságán.

4.6. Az osztályok grafikai analízise

Az osztályok közötti kapcsolatok elemzése komplex feladat, gráfelméleti ismereteket igényel [8]. Az ügyfél által adott példákban olyan esetek találhatóak, ahol a kapcsolatok többtagúak lehetnek, több elágazó csomóponttal. Az esetek nagy részében a kapcsolat kéttagú, tehát egy egyszerű linkhez két elem kapcsolódik, de olyanra is van példa, hogy a kapcsolatban 8-9 elem szerepel. Tovább nehezíti az osztályok grafikai

analízisét, mikor egy makro link (amely felbontható atomi linkekre) különböző osztályokhoz, különböző mennyiségű linkekkel kapcsolódik.

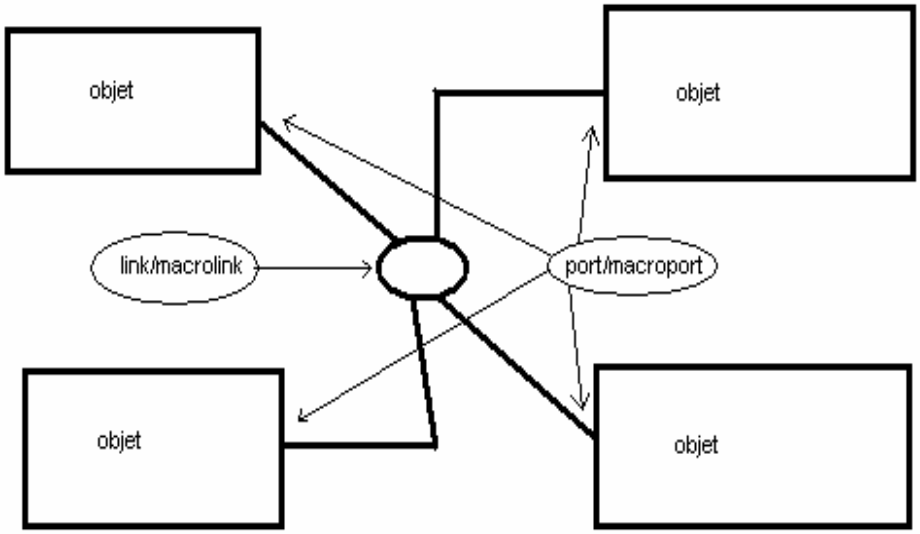


14. ábra: Példa osztályokra kapcsolatokkal (az ügyféljavaslatai alapján)

A 14 ábrán létrehozott modellel az a probléma, hogy nehézkes a megvalósítása. Így ábrázolva nagyon hamar áttekinthetetlenné válik egy többtagú elemből álló modell, és túl komplex az automatikus generálás.

Jgraph nagyon hatékony grafikus reprezentációs eszköz. Csomópontokat és kapcsolatokat ábrázol. Viszont nehéz megvalósítani az elemek közvetlen egymáshoz kapcsolódását. Innen jött az ötlet: szükség van egy úgynevezett **súlypont**ra. Ehhez a súlyponthoz kapcsolódnak az osztályok, ezáltal növekszik a generálási hatékonyság. Ráadásul, ha ezt a középpontot felruházzuk azonos képi objektum jogokkal, mint a többi osztályt, akkor az optimalizálási lehetőségek is nagymértékben javulnak. Ez az azonos jog abban mutatkozik meg, hogy a súlypont ugyanúgy szerepet vállal a kép optimalizációjában, vagyis a Jgraph csomag matematikai és gráfelméleti számításokat használó algoritmusaiban.

A 15. ábrán láthatjuk a módosított modellt.



15. ábra: A módosított modell

Első körben a módosított modellt építjük fel. Ez a modell segítségével töltsük fel elemekkel a grafikus megjelenítés adatbázisát.

4.7. Grafikus jelölésrendszer

Íme egy ajánlott grafikus jelölésrendszer az objektumok képre vitelének. Francia projekt lévén az elemek elnevezése specifikus, de a fontosabb elnevezések magyar megfelelői megtalálhatóak zárójelben.

A grafikus megjelenítéshez tartozó osztályokból képzett objektumok rendszerezése.

- 2 típusú objektum:
  - Nœud (csúcspont)
  - Lien (link)
- 6 típusú csúcspont :
  - Nœud 1 : macro-classe (makroosztály)
  - Nœud 2 : classe atomique (atomi osztály)

- Nœud 3 : port ou macro-port (port vagy makro port)
- Nœud 4 : lien simple (link)
- Nœud 5 : macro-lien (makro link)

- 2 type de liens (két típusú link)
  - Lien 1 : lien simple (egyszerű link)
  - Lien 2 : macro-lien (makro link)

- Kapcsolat típusok a csomópontok között:

	Nœud 1				
Nœud 1	Impossible	**Nœud 2**			
Nœud 2	Impossible	Impossible	**Nœud 3**		
Nœud 3	Impossible	Impossible	Impossible	**Nœud 4**	
Nœud 4	Lien 1	Lien 1	Lien 1	Impossible	**Nœud 5**
Nœud 5	Lien 2	Impossible	Lien 2	Impossible	Impossible

- A csúcspontok attribútumai.

Ezek különböző szín - méret - alak tulajdonságok.

		Nœud 1	Nœud 2	Nœud 3	Cas particulier	Nœud 4	Nœud 5
Objet	Forme	Rectangle	Rectangle	Rectangle		Arrondi	Arrondi
	Taille	80x60 pts	80x60 pts	Celle du nom		Celle du nom	Celle du nom
	Couleur	Bleu pâle	Jaune pâle	Rouge pâle		Bleu clair	Rouge clair
Contour objet	Style	Continu	continu	continu		continu	
					Si INPUT	pointillés	
	Epaisseur	3 pts	1 pt	2 pts		1 pts	3 pt
	Couleur	Noir	noir	noir		noir	
					Si REPORT	Rouge vif	
Nom générique de l'objet	Style	Gras	Normal	ND		ND	ND
	Taille	12	12	ND		ND	ND
	Couleur	Noir	Noir	ND		ND	ND
Nom spécifique de l'objet	Style	Italique	Italique	Italique encadré		Normal	Gras
	Taille	10	10	10		10	10
	Couleur	Noir	Noir	Noir		Bleu clair	Rouge

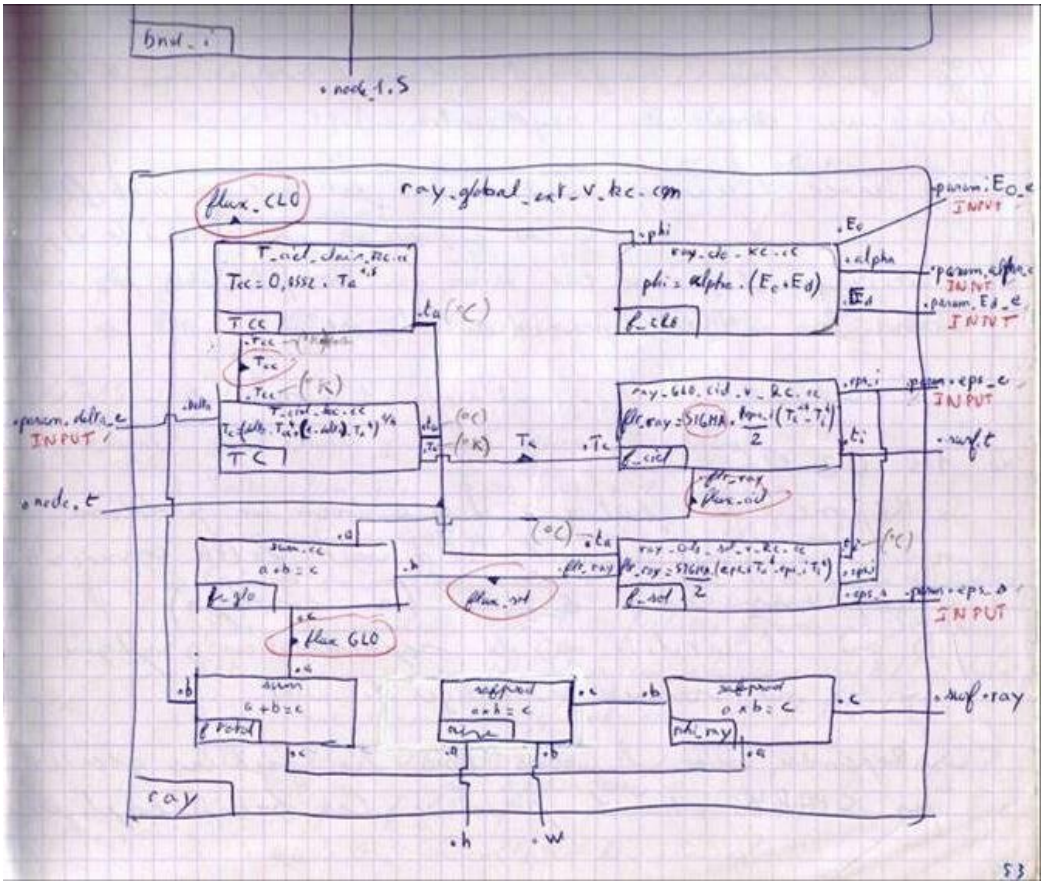
ND: Non Défini (nem definiált)

- A linkek attribútumai:

		Lien 1	Lien 2
Lien	Style	Continu	Continu
	épaisseur	1 pt	3 pts
	couleur	Bleu	Rouge
Extrémité côté arrondi	Style	Aucun	Aucun
Extrémité côté rectangle	Style	Carré plein	Rond plein
	Taille	7 pts	7 pts
Nom du port	Style	Normal	Gras
	Taille	10	10
	Couleur	Vert	Marron

5. Eredmények analízise

Bemutatom az eddig rendelkezésre álló módszereket az általam készített alkalmazás használata előtt. Ahogy a 16. ábrán is látható, a Spark programozók papír-ceruza módszert, vagy számítógépes szerkesztő segédeszközöket használtak. Nagyon komplex rendszereknél ez a módszer már nem kivitelezhető.



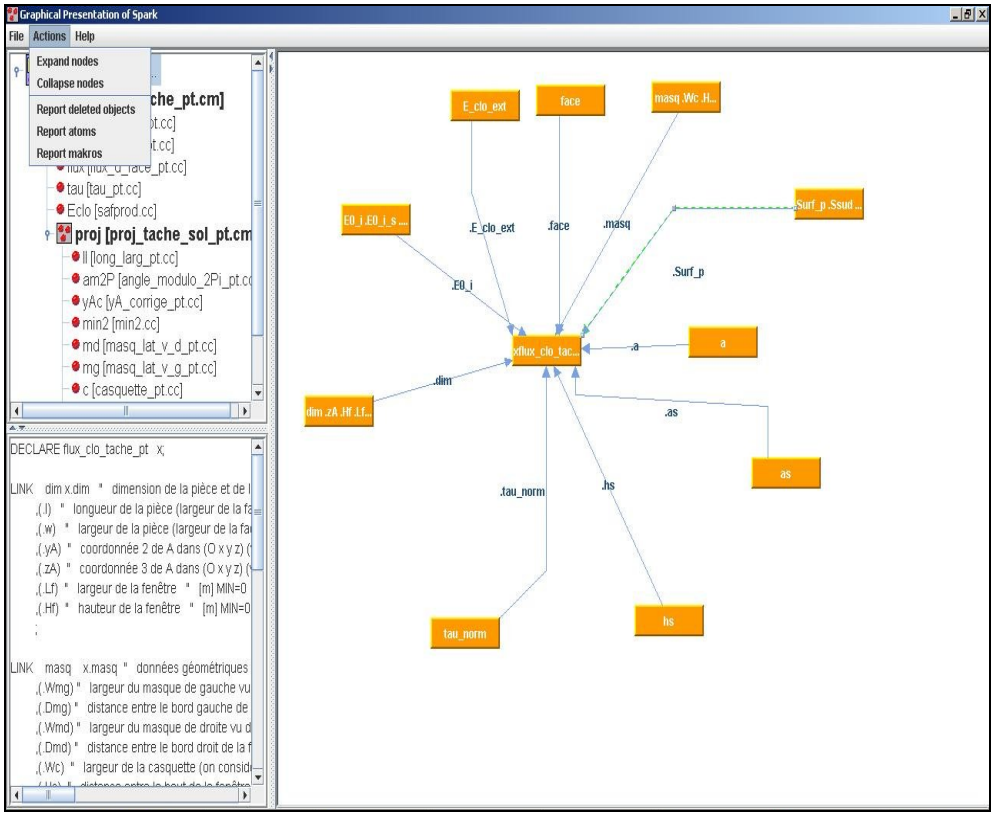
16. ábra: Eddig használt módszerek

A következőkben bemutatok néhány képet az általam készített alkalmazásról és az alkalmazásban elmentett képekről. Az alkalmazást az ügyféllel közösen ViewSpark-nak neveztünk el. A következőben bemutatott grafikus reprezentáción nincs alkalmazva a grafikus jelölésrendszer,

Ahogy láthatjuk a 17. ábrán, a megjelenített alkalmazás három zónára van osztva. Balra fenn láthatjuk az objektumfát. A makro osztályok négyszög jellel vannak megjelölve, az atomi osztályok pedig piros ponttal).

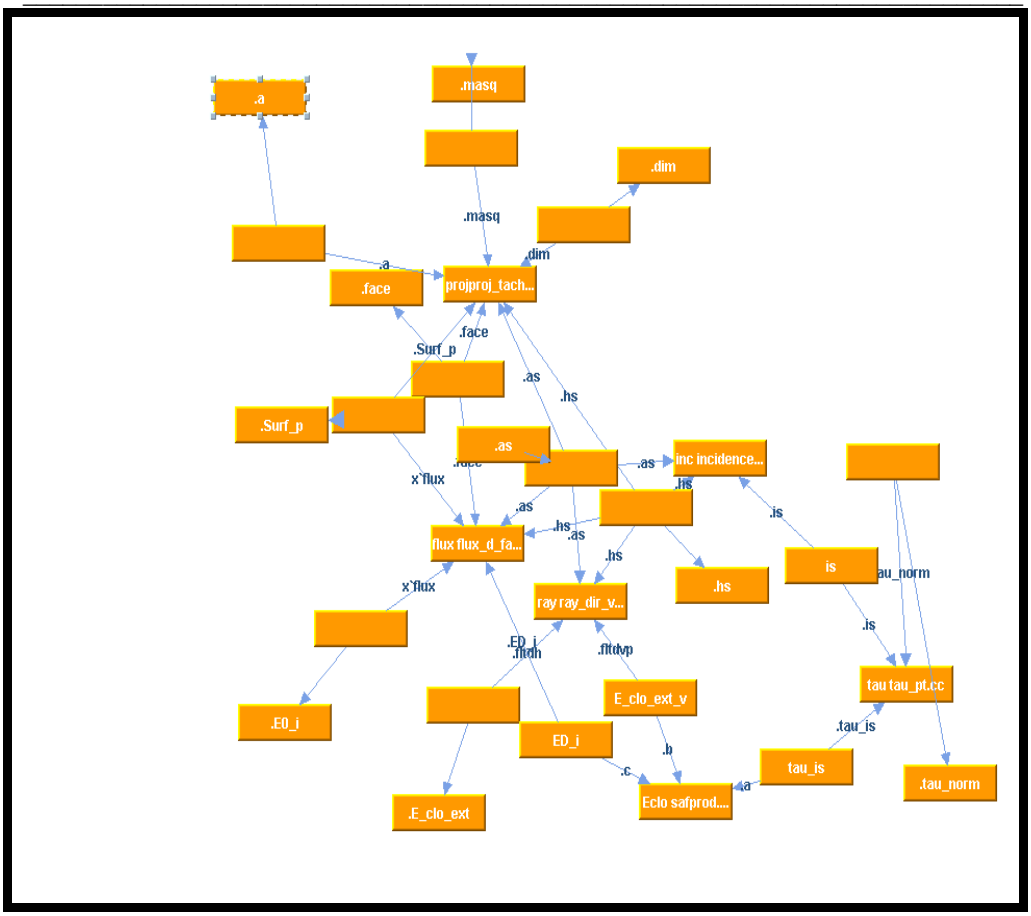
Mindhárom ablak tetszőlegesen méretezhető. Ezen felül a Spark programok forráskódja tetszőlegesen szerkeszthető. A frissített képhez újra kell generálni a .stp fájlt.

A kliens kérésére, mindhárom ablak tetszőlegesen menthető. Különféle kezelési lehetőségeket tartalmaz az objektumfa is az alkalmazásban.



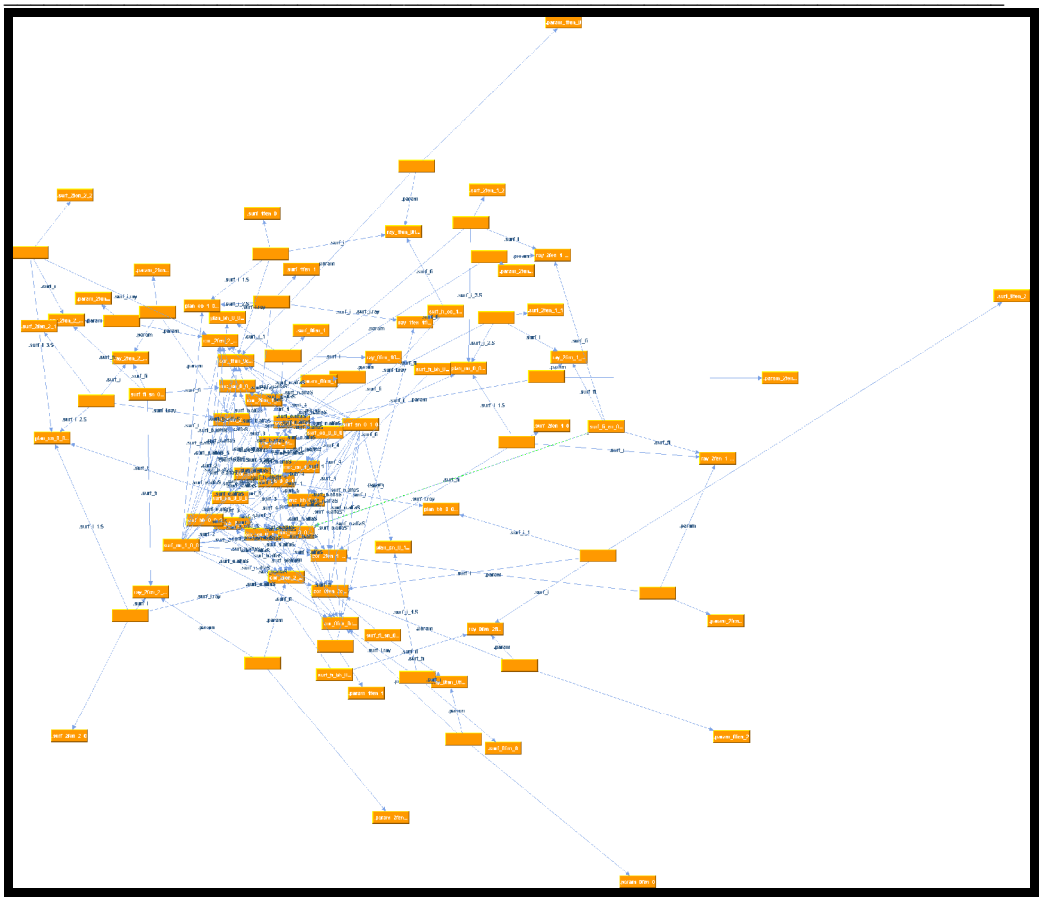
17. ábra: Teljes kép a ViewSpark szoftverről

A 17. ábrán láthatunk a ViewSpark egy több osztályból és linkből álló specifikusabb eredményét. Ez a kép már a szoftverből került lementésre. Ez egy generálás utáni alapállapot-pozíció.



18. ábra: Egy szimuláció és kapcsolatrendszerének a bemutatása

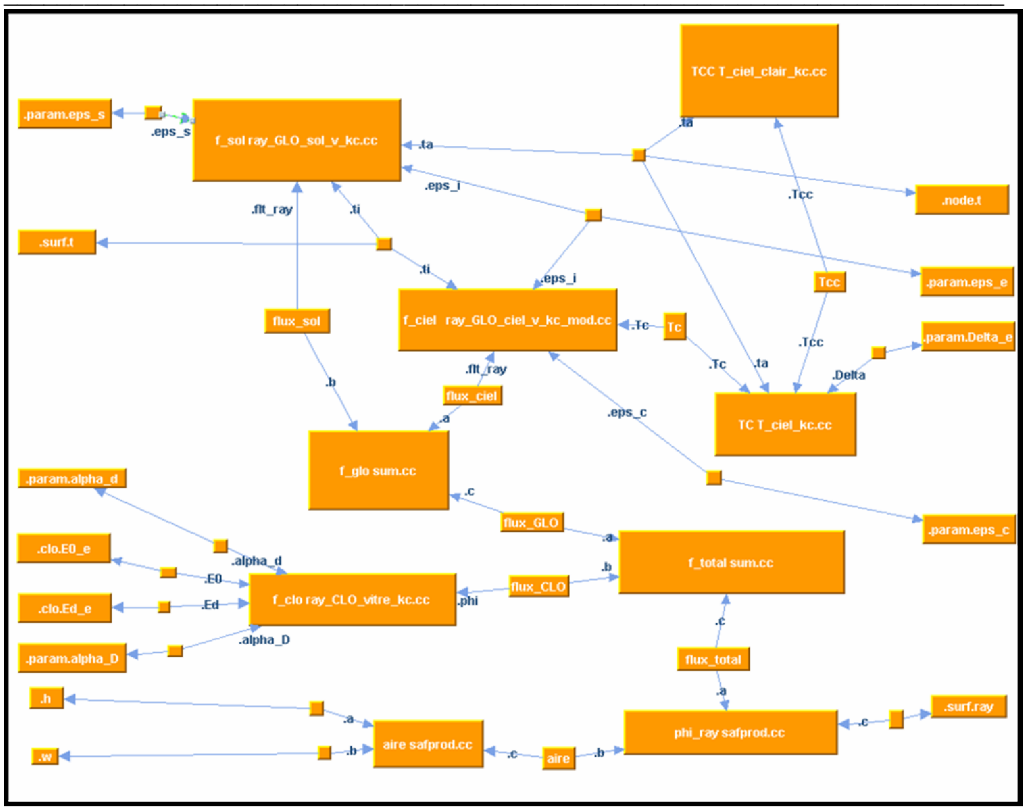
A készített alkalmazás a változtatott modell segítségével komplex és nagyszámú osztály ábrázolására ad lehetőséget. Ez az ábrázolás már alkalmasabb épületek energetikai szimuláció elkészítésének a segítésére.(18. ábra)



19. ábra: Mentett kép a ViewSpark szoftverből

Íme ahogyan a szoftver komplexebb feladattal is meg tud birkózni.

A szoftver segítségével a kirajzolt objektumok Drag&Drop technikával mozgathatók és a linkeket is tetszőlegesen lehet módosítani. A következő kép, amit a 19. ábrán láthatunk szintén a ViewSpark segítségével került mentésre.



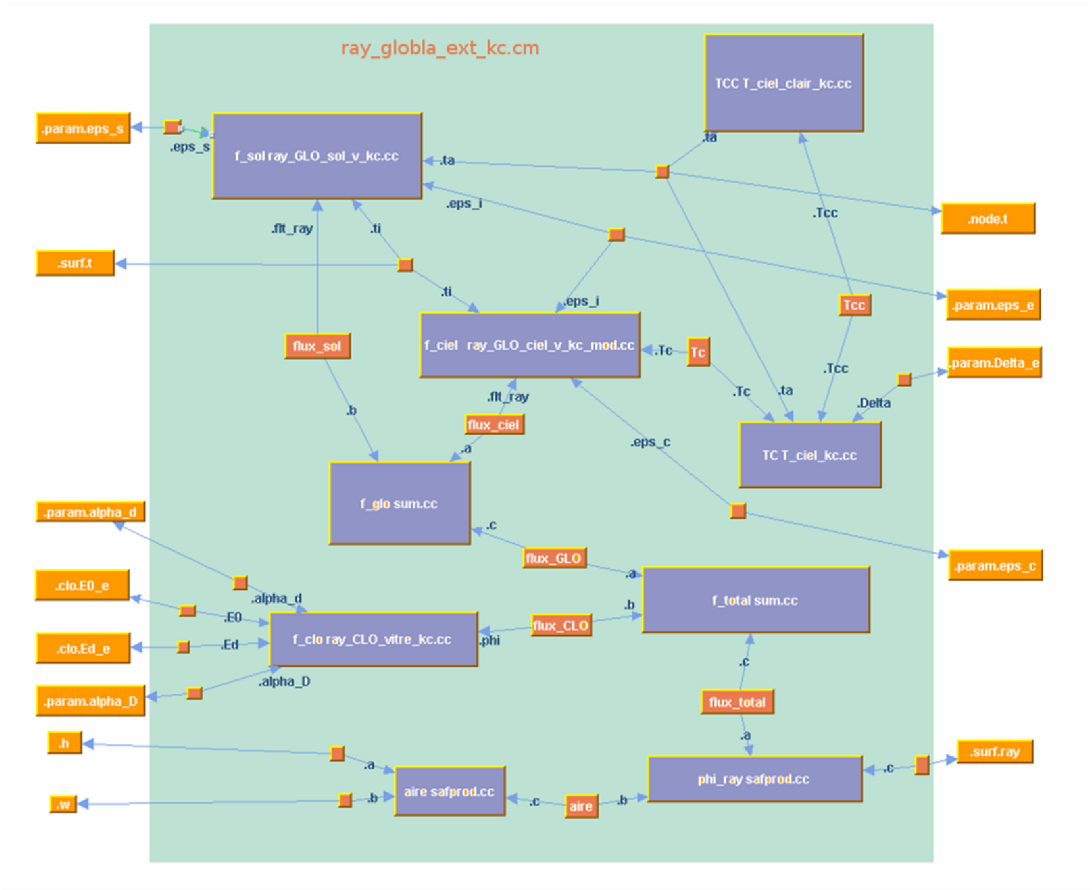
20. ábra: Drag &Drop és méretezés segítségével mentett kép

A 20 ábra már jobban tükrözi az elképzeléseket a program céljával kapcsolatban. Meghatározható minden egyes elemhez egy szerepkör, és ez a szerep határozza meg a képen látható elemek funkcióit.

6. Perspektívák

Számos fejlesztési pont és lehetőség van még a projektben. Mint ahogy a mentett képeken is látható, két típust különböztethetünk meg: a csúcspontokat és linkeket. A programban minden egyes csúcspont tudja a saját szerepét. Fogalmazhatnánk úgy is, hogy az objektumorientált szemlélet szerint minden egyes csomópontnak van „öntudata”. Tisztában van azzal, hogy neki mint csomópontnak milyen jellegű szerepe van a grafikus megjelenítésben. Tulajdoníthatunk neki szín, méret és forma tulajdonságokat is, amelyeket a Spark programozók tetszőlegesen értelmezhetnek.

A 21. ábrán a színeket már egy egyszerű képszerkesztővel adtuk hozzá. Az „öntudat” képességgel és az ún. *rendering* technikával rendelhetünk további tulajdonságokat minden egyes csúcsponthoz. A rendering technika szerint végigveszünk minden egyes elemet, és a típusától függően rendelünk hozzá értékeket. Így állhat elő egy olyan grafikus reprezentáció, amelyen a felhasználónak már kevesebbet kell módosítania.



21. ábra: Képszerkesztő által módosított szimuláció

További megvalósítandó funkcionalitások:

- Grafikus jelölésrendszer alkalmazása
- Ha rákattintunk egy objektumra, kiemeli az összes hozzá tartozó linket az alkalmazás által megjelenített forráskódban.
- Automatikusan megmutatja a forráskódban talált elem helyét a grafikus reprezentáción.
- További előugró (popup) menük használata az alkalmazásban
- Ha rákattintunk egy makro linkre, mutassa meg az összes hozzá tartozó linket és csomópontot.
- Mutassa meg a változók tulajdonságait, ha fölémegyünk az egérrel.

- A változók újracsoportosíthatósága a tulajdonságai alapján (specifikus lekérdezések).

- A skálázhatóság megőrzése és a bővíthetőség hozzáadása (újabb típusú elemek definiálásának a lehetősége)

7. Értékelés

Részt vettem és betekintést nyertem egy nemzetközi kutatóközpont a l'Institut Nationale de l'Energie Solaire (Nemzeti Napenergia Intézet) életébe.

Projekt tapasztalatot nyertem, láthattam miképpen zajlik le egy projekt elejétől a végéig. Számomra sokat jelent, hogy megismertem az informatika alkalmazhatóságának egy másik területét. Megtanultam, hogy mit jelent egy olyan ügyféllel dolgozni, akinek nem az informatika a szakterülete. A projekt nagyon sokat segített a francia nyelv gyakorlásában, minden problémát az ügyféllel és az informatikusokkal franciául kellett megvitatni. Megismertem egy másik programozási nyelvet, amelynek a koncepciója és a célja nagyon különbözött az általam ismertektől, ez pedig szélesítette a látókörömet.

A másik fontos dolog számomra a munkám eredményessége. Az általam megírt ViewSpark programot használják a Sparkkal dolgozó kutatók. Használatával időt takarítanak meg, könnyebben megértik a mások által megírt komponenseket, modelleket, szimulációkat. A projekt alatt sokat dolgoztam a dokumentáción. A fejlesztés most is folytatódik. A Nemzeti Napenergia Kutató Intézet a Savoie Egyetem Informatikai és Matematikai részlegével karöltve folytatja a már megkezdett munkát.

8. Irodalomjegyzék

[1] Mora, L. *«Prédiction des performances thermo-aérauliques des bâtiments par association de modèles de différents niveaux de finesse au sein d'un environnement orienté objet».* Thèse de doctorat. (2003)

[2] Dimitri Curtil *«Theoretical Computational Speed-Up in SPARK». Simulation Research Group at Lawrence Berkeley National Laboratory (2003)*

[3] SPARK 2.0 Reference Manual: Lawrence Berkeley National Laboratory. http://simulationresearch.lbl.gov/VS201/doc/SPARKreferenceManual.pdf (2003)

[4] Jgrapht referencia: http://jgrapht.sourceforge.net/

[5] David Benson *«JGraph and JGraph Layout Pro User Manual».* http://www.jgraph.com/pub/jgraphmanual.pdf (2007)

[6] EMF referencia: http://www.eclipse.org/modeling/emf/docs/

[7] Merlin referencia: «Runtime *Java User Interfaces Generator».* https://merlin.dev.java.net/servlets/ProjectDocumentList (2006)

[8] Chris K. Caldwell: *«Graph Theory Tutorials».* http://www.utm.edu/departments/math/graph/index.html *(1995)*